RAG as a collapsed NLG pipeline

Adarsa Sivaprasad, Barkavi Sundararajan, David M Howcroft
Department of Computer Science
University of Aberdeen, UK

Abstract

The NLG pipeline of Reiter and Dale has long served as the foundational framework for data-to-text system design and evaluation. However its relationship to modern generative architectures remains underexplored. In this conceptual analysis, we argue that Retrieval-Augmented Generation (RAG) constitutes a collapsed and partially reconstructed instantiation of the classical NLG pipeline, using it to identify failure modes of RAG around context faithfulness and retrieval non-determinism.

1 Background and motivation

The Natural Language Generation (NLG) pipeline proposed by Reiter and Dale (1997) has been the dominant conceptual framework for the design and development of data-to-text systems for over two decades. By decomposing data-to-text generation into the discrete modular stages of document planning, microplanning and surface realisation, the pipeline provided both a principled architecture for system builders and a shared vocabulary for researchers, structuring how data-to-text systems were designed and analysed (Reiter, 2025). Its influence is evident in systems across application domains, such as SumTime (Sripada, 2003), which employed explicit data interpretation and document planning for weather-forecast generation, and BT-45 (Portet et al., 2007) for clinical text summarisation. The pipeline shaped data-to-text generation by framing generation as a sequence of explicit decisions about content selection, document structuring, and linguistic expression.

The emergence of neural architectures for language generation shifted the field away from this modular paradigm. Sequence-to-sequence models (Sutskever et al., 2014) and later transformer-based architectures (Vaswani et al., 2017) enabled end-to-end generation without explicitly encoding content selection, planning, and surface realisation. Rather than eliminating these pipeline functions, neural architectures leave them implicit, distributing the corresponding decisions across learned representations. While some work has explored explicitly learning to make these decisions in neural networks (e.g., Puduppully et al., 2019), most recent work has built on transformer models without architectural changes designed to capture these aspects of the classical pipeline.

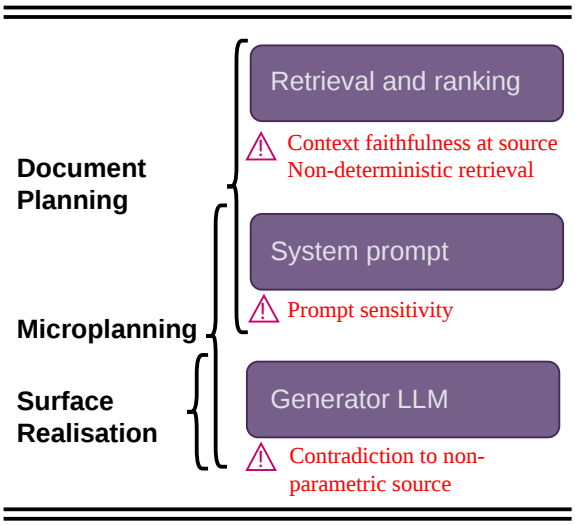

Figure 1: Classical NLG pipeline stages (left) and the corresponding RAG components (right), including overlap between document planning and microplanning. Each of the three RAG components is associated with particular challenges, highlighted below each component.

The advent of large-scale generative language models in 2020 (Brown et al., 2020) marked a further shift from both the symbolic pipeline and earlier neural architectures toward generation based on (large) language models ((L)LMs). While these models exhibit remarkable performance across a wide range of generation tasks, their reliance on parametric knowledge encoded during pre-training renders them susceptible to factual errors and confident confabulation, widely termed hallucination

Proceedings of the 1st Symposium on Natural Language Generation Evaluations, pages 33–38

(Ji et al., 2023). Retrieval-Augmented Generation (RAG), (Lewis et al., 2020), emerged as a widely adopted architectural response to this limitation. By grounding text generation on external, curated, non-parametric knowledge sources, RAG substantially reduces hallucination (Shuster et al., 2021) and has become a dominant framework for deploying LLMs in knowledge-intensive tasks.

In this work, we reflect on RAG through the lens of the classical NLG pipeline, analysing how its components align with — and diverge from — the classical NLG pipeline stages of data interpretation, document- and micro-planning, and surface realisation. We argue that RAG implicitly implements several features of the NLG pipeline in novel forms (summarised in Figure 1), with document planning taking place across the retrieval, ranking, and prompting stages of a RAG system, microplanning overlapping with prompting and generation from LLMs, and surface realisation fully limited to the LLM generation stage. The analogy also highlights the fundamentally new concerns in a RAG around retrieval quality, context faithfulness, and conflicts between retrieved content and the generator's parametric knowledge (Longpre et al., 2021), that fall outside the original pipeline's scope. By mapping this correspondence, we aim to illuminate both what has been recovered and what has been lost in the transition from symbolic to retrieval-augmented generation and to explore the implications for evaluation, interpretability, and system design in contemporary NLG.

2 The Classical NLG Pipeline

The classical NLG pipeline of Reiter and Dale (1997) decomposes generation into three main stages: *document planning*, *microplanning*, and *surface realisation*. Later data-to-text architectures extended this framework with earlier stages, such as signal analysis and data interpretation, to handle raw data inputs (Reiter, 2007, 2025). We consider the original three-stage pipeline to be closer to how RAG systems are deployed. Signal analysis and data interpretation as pre-cursor stages to document planning are designed to take masses of raw data and convert them into meaningful units, while the retrieval index of a RAG system is more like the database or collection of facts which results from such analyses: they provide the materials from which the document can be planned based on a given user query.

We illustrate these stages using SumTime (Sripada, 2003), a rule-based NLG system that generates marine weather forecasts from numerical weather prediction (NWP) data (e.g., wind speed and direction, temperature, pressure) to support offshore operations. It uses rules derived from knowledge acquisition tasks such as corpus analysis, expert consultations, and think-aloud sessions with forecasters, to drive decisions at each stage (Sripada et al., 2004). An extract of SumTime output is reproduced from Reiter (2025):

Wind(10M): S 16–21 backing SSE 21–26 by mid afternoon, then veering S by early evening and SSW 18–23 by midnight.

In this example, *S*, *SSE*, and *SSW* denote wind directions on the standard 16-point compass, and *"backing"* and *"veering"* are change verbs to describe shifts in wind direction.

Document planning makes two key decisions: *content selection* (which events to communicate) and *document structuring* (how to organise them into a coherent narrative) (Reiter, 2025). In SumTime, document structuring follows the forecast structure recommended by Weathernews UK, while content selection uses a bottom-up segmentation algorithm to group adjacent readings and identify meteorologically significant wind states and direction changes to mention (Sripada et al., 2003, 2004). These selected events are then ordered chronologically, as in the example above.

Microplanning determines how the document plan is expressed, including *lexical choice*, *referring expression generation*, and *aggregation*. In SumTime, corpus-derived rules guide the selection of domain-specific forecast verbs, such as *"backing"* and *"veering"* for wind-direction changes and *"increasing"* for wind-speed changes (Reiter, 2025). The rules also map time steps to time expressions such as *"by mid afternoon"* and *"by midnight"* (Sripada et al., 2002).

Surface Realisation renders the microplan as grammatically correct text, handling syntax, morphology, and punctuation (Gatt and Reiter, 2009).

3 RAG as a Collapsed Pipeline

A RAG system combines the parametric knowledge encoded in a pre-trained neural language model with a document index containing supple-

mental information which is called *non-parametric* as it is not encoded in the LMs trained parameters (Lewis et al., 2020). The non-parametric knowledge source is encoded in dense vector representations to enable easy retrieval based on distributional semantics. These representations capture meaning in a vector space, similar to the 'semantic' representations learned by neural LMs.

The LM combines retrieved non-parametric knowledge with parametric knowledge guided by a *prompt* which is a natural language specification of what the resulting text should look like, with or without examples (in few-shot and zero-shot prompting, respectively). In this analysis, we examine each of these components of a RAG system to characterise the data-to-text functions they perform.

3.1 Retrieval and reranking

Domain knowledge, typically in the form of documents, is represented as a collection of embeddings which implicitly encode the semantic content of the underlying document. Retrieval selects content from this indexed knowledge source based on a *user query*, potentially reranking this content by relevance (Glass et al., 2022), is therefore functionally analogous to content selection in the document planning module (Reiter and Dale, 1997).

However, classical document planning can be deterministic and is fully controllable when it is rule-based. The content selection is purposive and input is interpreted with respect to a specific generation task. In RAG, encoding is performed offline and independently of any particular query – the same representation must serve all possible future retrieval contexts. Hence, while a non-parametric source created for a specific task can be faithful to its context, general-purpose knowledge sources are susceptible to semantic ambiguity, since embeddings constructed without a specific generation intent cannot anticipate the full space of queries they will be expected to serve. Following Es et al. (2024), we term this *context faithfulness at the source* – a faithfulness risk structurally absent from the classical pipeline.

Further, document structuring in the classic NLG pipeline is governed by explicit communicative goals, and can be evaluated by comparing system outputs against expert-authored texts (Sripada, 2003). Retrieval, by contrast, is generally probabilistic and less deterministic. In the absence of an exact semantic match for a query, the retriever may select a misaligned yet closest match, introducing uncertainty into the content selection process. Uncertainty may also arise from the contradiction of the retrieved contents with the parametric knowledge of the generator LLM (Longpre et al., 2021). Since relevance scores do not guarantee factual alignment, the document planning stages of RAG require dedicated retrieval quality assessment.

3.2 System prompt

Prompt engineering has become an integral component of generative AI workflows, and, within RAG specifically, the prompt provides instructions influencing content aggregation and specificity, and specifying how to align to the communicative goal of the response (Schulhoff et al., 2024), ultimately overlapping with both document and micro-planning decisions.

This correspondence, however, remains underexamined. Unlike the retrieval stage, where the functional analogies to document planning is relatively direct, the prompt and the generator LLM relationship is tightly coupled. The prompt does not operate independently: its effect on aggregation and lexical choice is contingent on the instruction-following behaviour of the specific generator model it addresses, making it difficult to isolate prompt-level microplanning from the broader generative behaviour of the LLM. Further, different prompting strategies — zero-shot, few-shot, or chain-of-thought, may exert different influences on how the retrieved content is aggregated and expressed.

3.3 Generator LLM

Surface realisation in the classical NLG pipeline is the final stage, responsible for converting abstract linguistic representations into grammatically well-formed, fluent text. Within a RAG architecture, the LLM generator occupies this role.

Neural language models collapse the distinct stages of the NLG pipeline into a single learned process. Puduppully et al. (2019) demonstrate that neural models can encapsulate functions such as content selection, aggregation, and surface realisation within unified parameter spaces, for example. RAG can therefore be understood as a deliberate architectural counter-move. By externalising data interpretation, content planning and expression planning (prompting) into discrete upstream stages, it partially reconstructs the modularity that end-to-end neural generation had collapsed, constraining the LLM to focus on what it demonstrably does

NLG Pipeline Stage	Classical System *(Weather forecasts)*	RAG System *(Suggesting activities based on weather)*	Key Divergence
Document planning *(Content selection & structuring)*	Bottom-up segmentation selects significant wind states and direction changes. Selected events are ordered chronologically in the forecast text.	Probabilistic retrieval of activity descriptions based on ideal weather conditions provides high level structure in the LLM prompt.	Retrieval is non-deterministic with relevance scores which do not guarantee factual alignment.
Microplanning *(Lexicalisation & aggregation)*	Corpus-derived rules choose forecast verbs such as *"backing"*, *"veering"*, and *"increasing"*, and map time steps to expressions (*"by midnight"*).	Prompt guidance for what kind of register to use, influencing word and aggregation choices.	Prompt and LLM coupling means microplanning cannot be isolated from surface realisation.
Surface realisation	Realiser renders the forecast in domain-standard form, including syntax, punctuation, and formatting, e.g., *"S 16–21 backing SSE 21–26 by mid afternoon"*.	Generator LLM producing fluent activity suggestions based on on retrieved passages and given prompt.	LLM generation combines surface realisation with content and expression decisions.

Table 1: Highlighting similarities and differences between classical data-to-text pipeline systems like SumTime for weather forecasts (Sripada, 2003) and a potential RAG application in an related domain (suggesting activities appropriate for given weather conditions) highlighting differences between a classical pipeline application and a RAG application.

best, namely producing fluent, coherent, grammatically correct text conditioned on a structured input context. However, as a probabilistic model, the RAG pipeline accumulates different uncertainties discussed earlier at each stage. Further, the contradiction of retrieved information with the learned parametrised knowledge of the generator LLM is an added risk (Longpre et al., 2021).

3.4 Implications for Evaluation

The pipeline decomposition reveals where RAG can fail, and also informs how to detect those failures. We summarise evaluation criteria in RAG, motivated by classic NLG stage-specific failures :

- Contextual faithfulness check at retrieval to ensure appropriate content is available in knowledge source .
- Conduct explicit retrieval performance evaluations.
- Test the sensitivity of how the document plan is expressed and handled on the prompting strategy. This must include stability of output, output length and meaning preservation (Schulhoff et al., 2024).
- Quantifying uncertainty in surface realisation due to the contradiction of retrieved context and underlying LLM training knowledge, such as proposed in (Longpre et al., 2021).
- Granular error annotation addressing microplanning and surface errors, such as done (Thomson and Reiter, 2020; Sundararajan et al., 2025), that account for both lexical choices and aggregation decisions.

4 Discussions and Conclusion

By showing the analogy of RAG to the classic data-to-text pipeline, we illustrate that the architecture constrains generative LLMs, in which the model is

implicitly responsible for all pipeline stages simultaneously and where hallucination and factual drift are consequently most acute. As a surface realiser, unlike classical surface realisers, which operate on verified, structured linguistic representations, we note that the LLM generator lacks an intrinsic mechanism to detect or reject factually inconsistent, uncertain retrieved content. Faithfulness to the retrieved context is therefore not guaranteed by the architecture itself, but must be enforced through additional evaluation. The classical NLG pipeline has been applied across a wide range of data-to-text use cases and input modalities, including time series (Sripada et al., 2003), relational tables (Puduppully et al., 2019) and semantic triples (Gardent et al., 2017), which may call for additional steps of signal analysis or data interpretation. In this work, we abstract away from the input modality and focus on the three core generative stages that directly implicate the limitations of the RAG architecture.

We acknowledge that this analysis is primarily conceptual in nature and that the analogies are grounded in the literature rather than empirically validated. Future quantitative work will need to explore task-specific strategies for explicitly encoding classical NLG pipeline stages into RAG, such as prompting-based approaches of few-shot or chain-of-thought, and pipeline-informed retrieval strategies. Our analysis suggests that the NLG pipeline retains significance as a framework for understanding neural architecture, especially RAG, and is a diagnostic tool to identify where it can fail.

Acknowledgments

We thank Ehud Reiter for insightful discussions on the NLG pipeline and LLMs, which helped sharpen the analysis presented here. We thank the anonymous reviewers for their helpful comments, which have improved our paper, and for their encouragement to expand this work in the future. DMH was supported by CRUK grant EDDPJT-May23/100001.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:1877–1901.

Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. 2024. Ragas: Automated evaluation of retrieval augmented generation. In *Proc. of the 18th Conference of the European Chapter of the Association for Computational Linguistics (EACL): System Demonstrations*, pages 150–158.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. Creating training corpora for NLG micro-planners. In *Proc. of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 179–188, Vancouver, Canada. Association for Computational Linguistics.

Albert Gatt and Ehud Reiter. 2009. SimpleNLG: A realisation engine for practical applications. In *Proc. of the 12th European Workshop on Natural Language Generation (ENLG)*, pages 90–93, Athens, Greece. Association for Computational Linguistics.

Michael Glass, Gaetano Rossiello, Md Faisal Mahbub Chowdhury, Ankita Naik, Pengshan Cai, and Alfio Gliozzo. 2022. Re2g: Retrieve, rerank, generate. In *Proc. of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 2701–2715.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12).

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:9459–9474.

Shayne Longpre, Kartik Perisetla, Anthony Chen, Nikhil Ramesh, Chris DuBois, and Sameer Singh. 2021. Entity-based knowledge conflicts in question answering. In *Proc. of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7052–7063.

François Portet, Ehud Reiter, Jim Hunter, and Somayajulu Sripada. 2007. Automatic generation of textual summaries from neonatal intensive care data. In *Conference on Artificial Intelligence in Medicine in Europe*, pages 227–236. Springer.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019. Data-to-text generation with content selection and planning. In *Proc. of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6908–6915.

Ehud Reiter. 2007. An architecture for data-to-text systems. In *Proc. of the 11th European Workshop on Natural Language Generation (ENLG)*, pages 97–104.

Ehud Reiter. 2025. *Natural Language Generation*. Springer.

Ehud Reiter and Robert Dale. 1997. Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87.

Sander Schulhoff, Michael Ilie, Nishant Balepur, Konstantine Kahadze, Amanda Liu, Chenglei Si, Yinheng Li, Aayush Gupta, HyoJung Han, Sevien Schulhoff, and 1 others. 2024. The prompt report: A systematic survey of prompt engineering techniques. Preprint, arXiv:2406.06608.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803.

Somayajulu Sripada. 2003. SumTime-Mousam: Configurable marine weather forecast generator. *Expert Update*, 6(3):4–10.

Somayajulu Sripada, Ehud Reiter, Jim Hunter, and Jin Yu. 2002. Sumtime-meteo: Parallel corpus of naturally occurring forecast texts and weather data. Technical report, Computing Science Department, University of Aberdeen, Aberdeen, Scotland, Tech. Rep. AUCS/TR0201.

Somayajulu G. Sripada, Ehud Reiter, Ian Davy, and Kristian Nilssen. 2004. Lessons from deploying nlg technology for marine weather forecast text generation. In *Proc. of the 16th European Conference on Artificial Intelligence*, ECAI'04, page 760–764, NLD. IOS Press.

Somayajulu G Sripada, Ehud Reiter, Jim Hunter, and Jin Yu. 2003. Generating english summaries of time series data using the gricean maxims. In *Proc. of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 187–196.

Barkavi Sundararajan, Somayajulu Sripada, and Ehud Reiter. 2025. Input matters: Evaluating input structure's impact on LLM summaries of sports play-by-play. In *Proc. of the 18th International Natural Language Generation Conference*, pages 795–809, Hanoi, Vietnam. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 27.

Craig Thomson and Ehud Reiter. 2020. A gold standard methodology for evaluating accuracy in data-to-text systems. In *Proc. of the 13th International Conference on Natural Language Generation (INLG)*, pages 158–168, Dublin, Ireland. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems (NeurIPS)*, 30.